# Designing with NXP i.MX8M SoC

## Course Description

Designing with NXP i.MX8M SoC is a 3-days deep-dive training to the latest NXP application processor family.

The first part of the course starts by overviewing the i.MX8M Family and its Target applications, Device architecture (Quad/Dual/Dual-Lite), and family roadmap (i.MX8/8X).

The course then overviews various i.MX8M features such as video & graphics capabilities, audio capabilities, Cortex-A53, Cortex-M4, multicore, boot sequence, debug, system bus and interconnect, memory system (L1 cache, L2 cache, OCRAM, ROM, DDR controller), NAND/NOR, SD/eMMC, interrupts, DMA, fuses, pin mux and GPIO.

The course then introduces the connectivity in i.MX8M such as PCIe, USB, and other general-purpose connectivity (SPI, i2C, UART, PWM, GPT, Ethernet and GPIO).
Clock, PLL, power architecture, as well as debug and security are also introduced.

The second part of the course goes into advanced topics such as multicore configurations (AMP and SMP), the MU (Message Unit) and Sema4 (Semaphore) in order to communicate between Cortex-A53 and Cortex-M4, Interrupts partition strategy, and the boot process in multicore including secure boot. Power saving techniques are also covered in this section.

The final part focus on i.MX8M multimedia features, covering the display controller, GPU, and VPU.

The training uses Variscite's SoM: VAR-SOM-MX8.
Yocto project for the SoM is covered and hands-on labs provides the participant a way to experience with this new platform running applications and building Yocto project from source code, as well as debugging the system with Eclipse.

## Course Duration

3 days

## Goals

1. Become familiar with i.MX8M architecture
2. Become familiar with Heterogenous ARM cores: Cortex-A53 and Cortex-M4
3. Design SMP and AMP efficient platforms
4. Become familiar with i.MX8M multimedia, graphics and audio capabilities
5. Become familiar with clock, reset, power management and debug strategy
6. Become familiar with i.MX8 memory architecture and capabilities
7. Become familiar with i.MX8M Boot process including secure boot
8. Become familiar with i.MX8M hardware SoM
9. Use Yocto project to run applications and build your Linux Kernel from source code, and debug the project

## Target Audience

Software engineers that would like developing software and BSP for platforms based on i.MX8M SoC.

## Prerequisites

- Computer architecture background
- ARM architecture is an advantage but not mandatory
- Experience in developing embedded systems
- C/C++ knowledge

Approved Training Partner
ARM

HandsOn Training


# Day 1

## ➢ Introduction to the i.MX8M Family

- i.MX applications processor values
- i.MX processor portfolio
- i.MX8M main target applications
- i.MX8M key features
- i.MX8M Quad, Dual, DualLite and Tiny block diagrams and differences
- i.MX8M qualification levels and package type
- i.MX8M evaluation boards, SoM, and Software support

❖ **Lab #1: Board and Tools Bring-up**

## ➢ CPU Platform

- Cortex-A53 CPU platform overview
- Cortex-A53 versus Cortex-A9/Cortex-A7
- What's new in ARMv8-A
  - Privilege levels
  - A32 vs A64
  - AArch64 registers
  - A64 instruction set
  - AArch64 exception model
  - AArch64 memory model

- Software engineer's guide to the Cortex-A53 MPCore
  - Cortex-A53 pipeline
  - Branch prediction resources
  - Cache overview
    - L1 cache
    - L2 cache
  - Data cache coherency
  - Memory Management Unit (MMU)
  - Other micro-architecture features
  - Interrupt and bus interfaces
  - Debug and timers
  - Power management
  - Clocking
  - NEON
  - GIC
  - PMU

❖ **Lab #2: Measuring Application Performance, with Multiple Interrupts on a Single Core using the PMU**

- Cortex-M4 platform
  - Cortex-M4 features
  - M-profile instructions
  - Core register set
  - Processor pipeline
  - Cycle counting
  - Memory map
  - Bitwise memory access
  - Bit banding
  - Modes privilege and stacks
  - Interrupts and exceptions
  - Memory Protection Unit (MPU)
  - Tightly Coupled Memory (TCM)
  - Cache features
  - Processor core, space and Backdoor port accesses
  - SRAM accesses
  - Power management
  - Core debug
  - System timer
  - Floating point unit
  - Cortex-M4 use case

❖ **Lab #3: Configure the MPU with Different Access Permissions and Identify Stack Overflow**

# Day 2

## ➢ i.MX8M Memory System Overview

- ❖ Internal memory (Cache, TCM, OCRAM, ROM, ROMCP)
- ❖ Nand/Nor flash storage
- ❖ SD/eMMC interface
- ❖ DRAM interface

## ➢ i.MX8M Messaging Unit (MU)

- ❖ How to communicate between Cortex-M and Cortex-A?
- ❖ Massaging versus interrupt
- ❖ MU main features
- ❖ Processor A side and B side memory mapping
- ❖ MU programmer's model
- ❖ Passing messages examples
  - o Short messages
  - o Frame information
  - o Event notices and requests
  - o Fixed length data
  - o Announcements
- ❖ MU in low power modes
- ❖ Event update timing
- ❖ MU interrupts
- ❖ Interrupt messaging protocols
- ❖ Exclusive access to shared memory
- ❖ DMA and MU
- ❖ Software restrictions when accessing the MU

- ❖ **Lab #4: Configure the MU to Send Messages Between a Cortex-M4 and Cortex-A53**

## ➢ i.MX8M Clock Management

- ❖ Clock Control Module (CCM) overview
- ❖ CCM block diagram
- ❖ PLLs
- ❖ Crystal Oscillator (XTALOSC)
- ❖ Clock roots, max frequency and source select
- ❖ CCM output clock connectivity and gating

- Clock divider
- Clock switching multiplexer
- Clock gate
- Clock slices
- Access control
- System level considerations
- Programmer's guide

## i.MX8M System Reset Controller (SRC)

- SRC overview
- Reset and power-up sequence
- External POR
- Internal POR
- Reset inputs & outputs
- Parallel reset requests
- Boot mode control

## i.MX8M Power Management

- General Power Controller (GPC) overview
- GPC main features
- GPC block diagram
- Processors modes (RUN, Low power, WAIT, STOP, Deep Sleep)
- Low power mode process (entering and exiting)
- Power Gating Controller (PGC) overview
- PGC power domains
- Triggering the PGC via HW and SW
- Power control for A53 platform
- Power control for the M4 platform
- Thermal Management Unit (TMU)

- **Lab #5: Power Consumption Control with Various Methods (SMP and AMP modes)**

## i.MX8M DDR Controller (DDRC)

- Device supported and maximum capacity
- DDRC block diagram
- DDRC main features
- AXI Port Interface (XPI) block
  - Read address channel
  - Write address channel
  - Read data and response channel
  - Write data channel
  - Data width conversion

- Write response channel
- Port Arbiter (PA) block
- DDR Controller (DDRC) block
- APB Register block
- Exclusive access support
- Software coherency for AXI ports
- AXI data channel interleaving
- Early burst termination (EBT)
- ECC error

## i.MX8M SDMA

- SDMA overview
- SDMA block diagram
- SDMA main features
- SDMA Core
- SDMA Scheduler
- Burst DMA unit
- Peripheral DMA unit
- SDMA security support
- OnCE and PCU debug states
- SDMA clocks and low power modes
- SDMA APIs
- SDMA initialization
- SDMA programmer's model

- **Lab #6: Measuring the memcpy() Function Using SDMA, Cortex-A53, and Cortex-M4**

## i.MX8M Connectivity & Mass Storage Overview

- USB 3.0
- PCIe
- Ethernet MAC
- ECSPI
- Quad SPI
- uSDHC
- timers

## i.MX8M Boot

- System boot overview
- Boot modes
- Pin settings
- Boot sequence
- Boot security settings

- Boot eFUSE
- Device Configuration Data (DCD)
- Internal ROM/RAM memory map
- Boot block activation
- Clocks at boot time
- Enabling MMU & caches
- Exception and interrupt handling during boot
- Persistent bits
- Cortex-A and Cortex-M boot process
- Boot devices supported
- Program image
  - IVT
  - HDMI image boot-up
  - DCD
  - Plugin image
- Serial downloader
- Recovery devices
- USB low power boot
- High Assurance Boot (HAB)

- **Lab #7: Apply Cold and Warm Boot**

# Day 3

## i.MX8M Multimedia Overview

- Multimedia components
  - Enhanced LCD interface (eLCDIF)
  - GPU
  - HDMI Transmitter controller
  - MIPI_DSI
  - MIPI_CSI
  - Sony/Philips Digital Interface (SPDIF)
  - Synchronous Audio Interface (SAI)
- GPU fundamentals
  - What is a GPU?
  - Real time rendering and its importance
  - GPU types for i.MX8
  - GC7000 Lite architecture
  - Graphics pipeline front end
  - Unified shader engine: Vertex
  - 3D rendering engine

- Unified shader engine: Fragment
- Texture engine
- Pixel engine
- Vulkan support
- i.MX8M display pipeline
- i.MX8M DPR display controller cache
- GPU performance
- Supported APIs
- GPU debug tool

❖ Video Processing Unit (VPU)

- OpenMAX IL API
- Hantro HW decoder and post processor
- OpenMAX IL API functionality
- Decoder features
- Input & output buffers
- Video frame storage formats
- Interface functions

❖ Display Controller Subsystem (DCSS)

- DCSS major modes: HDR10, Dolby Vision
- Pixel processing
- Video only
- Graphics only
- Video with graphics overlay
- Picture in graphics (PIG)
- Picture in picture (PIP)
- Video with PIP and graphics
- DCSS memory map
- Dolby Vision mode operation
- HDR10 mode operation
- DCSS interrupts
- DCSS block control
- Display timing generator
- Context load
- Graphics decompression
- Decompression and tile to raster conversion
- Display, prefetch and resolve
- Scaler
- Look up table load
- HDR10 image processing
- Color sub-sampler
- Write scale and read surface

❖ **Lab #7: Build Video and Graphics Stream Flow to Display Controller**

❖ **Lab #8: Decode Two Video Streams and Display PIP on LCD**